

On the Linear Programming Bound for Lee-codes

Helena Astola, Ioan Tabus

Department of Signal Processing, Tampere University of Technology

P.O. Box 553, FI-33101 Tampere, Finland

Email: helena.astola@tut.fi

Abstract

Finding the largest code with a given minimum distance is one of the most basic problems in coding theory. In this paper, we study the linear programming bound for codes in the Lee metric. We introduce refinements on the linear programming bound for linear codes in the Lee metric, which give a tighter bound for linear codes. We also discuss the computational aspects of the problem and introduce a more compact program by the obtained refinements, and a recursive method for computing the so-called Lee-numbers, which are important in the computation of the linear programming bound.

1 Introduction

Lee-codes were first introduced by C. Y. Lee in [1]. The properties of Lee-codes and, in particular, the existence or nonexistence of perfect codes in the Lee metric have been studied by numerous authors, for example, in [2] and [3], and more recently in [4], [5], [6] and [7]. However, the research and literature on Lee-codes is not extensive. In data transmission, the Lee metric can be used in phase modulation schemes, since corrupted digits of phase-modulated signals are more likely to have only slightly different phase than greatly different phase compared to the original signal [8]. There have been some more recent applications of Lee-codes to, for example, VLSI decoders and fault-tolerant logic, which are discussed in [9], [10] and [11].

One of the most fundamental problems in coding theory is finding the largest code with a given minimum distance. The problem has been studied by several authors, in particular in the Hamming metric. The most well-known bounds are the Hamming bound, Plotkin bound, Singleton bound and Elias bound, and these bounds have been formulated for the Lee metric also. The Hamming bound states that the total volume of the radius t spheres

Date: June 13, 2014.

Keywords: Lee-codes, association schemes, linear programming bound, linear codes.

around codewords of a code with minimum distance $2t+1$ is at most the volume of the entire space. Codes that meet the Hamming bound are called perfect codes. The Plotkin bound is based on the observation that the minimum distance between any pair of codewords cannot exceed the average distance between all pairs of codewords. The Singleton bound is for linear codes and is based on the observation that the minimum distance of a code cannot be greater than the minimum distance of any of its subcodes. The Elias bound basically combines the Hamming bound and the Plotkin bound to obtain a stronger bound for medium rates, since the Hamming bound is tight at high rates and the Plotkin bound is tight at low rates.

In [12], Delsarte introduced association schemes to coding theory to deal with topics involving the distance distribution of a code. The theory and applications of association schemes into coding theory have been studied by numerous authors, and an extensive survey of these is given in [13]. An important approach to the problem of determining the upper bound for the size of a code is the linear programming approach, which follows from the association scheme structure in the Hamming metric [12], [14]. In fact, in the Hamming metric, the asymptotically best upper bound is the McEliece-Rodemich-Rumsey-Welch bound, see [15], which is based on the linear programming approach. The Lee metric also forms an association scheme, although the structure is more complicated. In the Hamming metric, the distance relations between codewords directly define an association scheme, but not in the Lee-metric. Therefore, based on the linear programming bound, it is not possible to formulate simple expressions for bounds in the Lee metric. Generalizing to finite Frobenius rings, the linear programming bound has been studied recently in [16].

For the Hamming metric, extensive tables of bounds on the size of codes for both binary and non-binary codes have been constructed, and such tables can be found, for example, in [17]. For the Lee metric, some values have been computed for small alphabets in [18]. In [19], a recursive formula for computing the Lee-numbers needed in the computation of the linear programming bound was introduced and more extensive tables with presently known best upper bounds can be found in [20].

In this paper, we discuss refinements on the linear programming bound for linear Lee-codes. In the Hamming metric, multiplying codewords by some constant does not change the weight of the codewords. However, in the Lee metric, multiplication typically changes the Lee-composition of the codeword and so also usually the Lee-weight. With linear codes, since they are linear subspaces of vector spaces, all the multiplied versions of any codeword also belong to the code. An important observation is that there must be as many codewords having the Lee-composition of a given codeword \mathbf{x} as there are codewords having the Lee-composition of $r\mathbf{x}$ that is obtained by multiplication of the given codeword \mathbf{x} by some constant r . Therefore, we get additional equality constraints between the coefficients of the weight distribution in the linear programming problem. Also, we can reduce the complexity of the problem since some coefficients can be assigned to zero following from the above observation. Furthermore, for a linear code, the linear transformation of the distribution vector by the second eigenmatrix of the scheme gives the distribution vector of the dual code. As it is also linear, the above observations also give constraints for the transformed vector, which turn out to be equivalent to the constraints given by the linearity of the code. By

introducing these equalities, we obtain a tighter bound for linear codes in the Lee metric.

The paper is organized as follows. Because the literature on Lee-codes is somewhat scattered and the notations vary, in Section 2 we review the concept of association schemes, in particular the Lee-scheme, and computing the linear programming bound for Lee-codes. Important concepts when determining the linear programming bound are Lee-compositions and Lee-numbers. In Section 3, we introduce the refinements on the linear programming bound for linear Lee-codes. In Section 4, we discuss the computational aspects in the linear programming problem. We discuss the computation of Lee-numbers, which can be done by recursion, and introduce an effective recursion based on the polynomial representation of the Lee-numbers. We also introduce a more compact linear program for solving the bounds for linear Lee-codes. We present the results obtained by applying the refinements in Section 5 together with some example linear codes that meet the obtained bounds.

2 Lee Codes and Lee Schemes

In this section, we give a short but careful review of codes and schemes in the Lee metric. For basic properties of error-correcting codes we refer to [8], [14].

Denote by \mathbb{Z}_q^n the set of n -tuples with elements from the set $\{0, 1, \dots, q-1\}$, i.e.,

$$\mathbb{Z}_q^n = \{\mathbf{x} = [x_1, \dots, x_n] \mid x_i \in \{0, 1, \dots, q-1\}\}.$$

The elements of \mathbb{Z}_q^n are q -ary vectors of block length n . A code C is a subset of \mathbb{Z}_q^n . When $q = p^k$, where p is prime, \mathbb{Z}_q^n is a vector space over the field of q elements, denoted by \mathbb{F}_q^n . In this paper, for simplicity, we discuss only fields \mathbb{F}_q^n , where q is prime and the Lee-distance is naturally defined on the elements. C is a linear code if it is a linear subspace of \mathbb{F}_q^n . The elements of C are called codewords. A linear code C of dimension $k \leq n$ is spanned by k linearly independent vectors of C .

The Hamming distance $d_H(\mathbf{x}, \mathbf{y})$ of vectors \mathbf{x} and \mathbf{y} of length n is the number of coordinates where \mathbf{x} and \mathbf{y} differ, i.e. $d_H(\mathbf{x}, \mathbf{y}) = |\{i \mid x_i \neq y_i\}|$. The Hamming-weight w_H of a vector \mathbf{x} is $w_H(\mathbf{x}) = d_H(\mathbf{0}, \mathbf{x})$.

The Lee distance $d_L(\mathbf{x}, \mathbf{y})$ of q -ary vectors \mathbf{x} and \mathbf{y} of length n is defined as follows:

$$d_L(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^n \min(|x_i - y_i|, q - |x_i - y_i|). \quad (1)$$

The Lee-weight w_L of a vector \mathbf{x} is $w_L(\mathbf{x}) = d_L(\mathbf{0}, \mathbf{x})$.

A code C is e -error-correcting if the minimum distance between two codewords is $2e + 1$. An e -Lee-error-correcting code will be able to detect and correct errors, which are at the Lee distance less than or equal to e from the encoded sequence, i.e., have Lee-weight less than or equal to e .

An association scheme is a set together with relations defined on it that satisfy certain properties. The following definitions are given according to [3], [12], [14]:

Definition 1. An association scheme with n classes consists of a finite set X together with $n + 1$ relations R_0, R_1, \dots, R_n defined on X which satisfy

- (i) Each R_i is symmetric: $(x, y) \in R_i \Rightarrow (y, x) \in R_i$.
- (ii) For every $x, y \in X$, $(x, y) \in R_i$ for exactly one i .
- (iii) $R_0 = \{(x, x) \mid x \in X\}$ is the identity relation.
- (iv) If $(x, y) \in R_k$, the number of $z \in X$ such that $(x, z) \in R_i$ and $(y, z) \in R_j$ is a constant c_{ijk} depending on i, j, k but not on the particular choice of x and y .

For example, the Hamming scheme consists of the set of q -ary vectors of length n and the vectors $\mathbf{x}, \mathbf{y} \in R_i$ if their Hamming distance is i . It is easy to verify, that the above conditions hold for the Hamming scheme.

In order to define the Lee-scheme, first we need to define the Lee-composition of a vector. The Lee-composition $l(\mathbf{x})$ of $\mathbf{x} \in \mathbb{Z}_q^n$ is the vector

$$l(\mathbf{x}) = [l_0(\mathbf{x}), l_1(\mathbf{x}), \dots, l_s(\mathbf{x})],$$

where $s = \lfloor \frac{q}{2} \rfloor$ and $l_i(\mathbf{x})$ is the number of the components of \mathbf{x} having Lee-weight i .

Now, consider the q -ary vectors of length $n = 1$. For this case, the distance relations define an association scheme in the Lee metric. More formally, let $X = \mathbb{Z}_q$ and define the relations R_0, R_1, \dots, R_s , $s = \lfloor \frac{q}{2} \rfloor$ with

$$(x, y) \in R_i \Leftrightarrow d_L(x, y) = i.$$

The conditions (i)-(iv) of an association scheme can easily be shown to be satisfied.

For $n > 1$ the Lee-scheme is defined as the Delsarte extension of the one-dimensional Lee-scheme and, thus, forms an association scheme defined as follows. Take two elements $\mathbf{x} = [x_1, \dots, x_n]$, $\mathbf{y} = [y_1, \dots, y_n]$ of \mathbb{Z}_q^n . Let $\rho_t(\mathbf{x}, \mathbf{y})$ be the number of integers i , $1 \leq i \leq n$ such that $(x_i, y_i) \in R_t$ and define the following $(s + 1)$ -tuple:

$$\rho(\mathbf{x}, \mathbf{y}) = [\rho_0(\mathbf{x}, \mathbf{y}), \rho_1(\mathbf{x}, \mathbf{y}), \dots, \rho_s(\mathbf{x}, \mathbf{y})].$$

Now $\rho(\mathbf{x}, \mathbf{y})$ equals the Lee-composition of the vector $\mathbf{x} - \mathbf{y}$. The number of distinct Lee-compositions is $\binom{n+s}{s}$.

Let $\rho^{(0)} = \rho(\mathbf{x}, \mathbf{x}) = [n, 0, \dots, 0]$, i.e., the Lee-composition of the all zero vector, and denote by $\rho^{(1)}, \dots, \rho^{(\alpha)}$, where $\alpha = \binom{n+s}{s} - 1$, the other distinct Lee-compositions. Let us define the set $K_0, K_1, \dots, K_\alpha$ of relations on \mathbb{Z}_q^n as follows

$$(\mathbf{x}, \mathbf{y}) \in K_i \Leftrightarrow \rho(\mathbf{x}, \mathbf{y}) = \rho^{(i)}.$$

In other words, $(\mathbf{x}, \mathbf{y}) \in K_i$ if the Lee-composition of the vector $\mathbf{x} - \mathbf{y}$ equals $\rho^{(i)}$.

Now \mathbb{Z}_q^n together with the relations K_i form an association scheme of α classes. This will be called the Lee-scheme.

The relations K_i can be described by their adjacency matrices, i.e., matrices D_i with rows and columns labeled by the points of \mathbb{Z}_q^n , where

$$(D_i)_{\mathbf{x}, \mathbf{y}} = \begin{cases} 1 & \text{if } (\mathbf{x}, \mathbf{y}) \in K_i, \\ 0 & \text{otherwise.} \end{cases}$$

These adjacency matrices generate an associative and commutative algebra called the Bose-Mesner algebra of the association scheme.

Let $\xi = \exp(\frac{2\pi\sqrt{-1}}{q})$. When $\mathbf{t} = [t_0, \dots, t_s]$ and $\mathbf{u} = [u_0, \dots, u_s]$ are Lee-compositions we define the *Lee-numbers* $L_{\mathbf{t}}(\mathbf{u})$ [3] from

$$\begin{aligned} & \prod_{l=0}^s (z_0 + (\xi^l + \xi^{-l})z_1 + (\xi^{2l} + \xi^{-2l})z_2 + \dots \\ & + (\xi^{sl} + \xi^{-sl})z_s)^{u_l} = \sum_{\mathbf{t}} L_{\mathbf{t}}(\mathbf{u}) z_0^{t_0} \dots z_s^{t_s}, \text{ for } q = 2s + 1 \end{aligned} \quad (2)$$

and from

$$\begin{aligned} & \prod_{l=0}^s (z_0 + (\xi^l + \xi^{-l})z_1 + (\xi^{2l} + \xi^{-2l})z_2 + \dots \\ & + (\xi^{(s-1)l} + \xi^{-(s-1)l})z_{s-1} + \xi^{sl}z_s)^{u_l} \\ & = \sum_{\mathbf{t}} L_{\mathbf{t}}(\mathbf{u}) z_0^{t_0} \dots z_s^{t_s} \quad \text{for } q = 2s. \end{aligned} \quad (3)$$

Consider the Lee-scheme \mathbb{Z}_q^n with the relations $K_{\mathbf{t}}$. Let C be a nonempty subset of \mathbb{Z}_q^n . The inner distribution of C is the $(\alpha + 1)$ -tuple of rational numbers $B_{\mathbf{t}}$, where

$$B_{\mathbf{t}} = \frac{1}{|C|} |K_{\mathbf{t}} \cap C^2|. \quad (4)$$

Now

$$B_{\mathbf{t}_0} = 1, \quad B_{\mathbf{t}} \geq 0 \text{ and } \sum_{\mathbf{t}} B_{\mathbf{t}} = |C|. \quad (5)$$

For Lee-compositions \mathbf{k}

$$B'_{\mathbf{k}} = \frac{1}{|C|} \sum_{\mathbf{t}} L_{\mathbf{k}}(\mathbf{t}) B_{\mathbf{t}} \geq 0, \quad (6)$$

i.e., certain linear combinations of the numbers $B_{\mathbf{t}}$ are nonnegative, which makes it possible to apply the linear programming bound to Lee-codes. This can be proved using the properties of the association scheme, see [3].

Let us write for a composition $\mathbf{t} = [t_0, t_1, \dots, t_s]$ [3]:

$$\begin{bmatrix} n \\ \mathbf{t} \end{bmatrix} = \binom{n}{\mathbf{t}} 2^{n-t_0} \quad \text{for } q = 2s + 1$$

and

$$\begin{bmatrix} n \\ \mathbf{t} \end{bmatrix} = \binom{n}{\mathbf{t}} 2^{n-t_0-t_s} \quad \text{for } q = 2s,$$

where $\binom{n}{\mathbf{t}}$ is the multinomial coefficient, which for nonnegative integers t_1, \dots, t_r with $m = t_1 + \dots + t_r$ is defined as

$$\binom{m}{t_1, \dots, t_r} = \frac{(t_1 + \dots + t_r)!}{t_1! \dots t_r!} = \frac{m!}{t_1! \dots t_r!}.$$

The value of $\begin{bmatrix} n \\ \mathbf{t} \end{bmatrix}$ corresponds to the number of vectors in \mathbb{Z}_q^n for which the Lee-composition is \mathbf{t} . Let C be a block code of length n over \mathbb{Z}_q . The inner distribution $[B_{\mathbf{t}_0}, \dots, B_{\mathbf{t}_\alpha}]$ of C in the corresponding Lee-scheme is given by (4). Due to (5) and since the eigenvalues of the Lee-scheme are given by the Lee-numbers we have

$$|C| = \sum_{i=0}^{\alpha} B_{\mathbf{t}_i}, \quad B_{\mathbf{t}} \geq 0, \quad \mathbf{t} \in \{\mathbf{t}_0, \dots, \mathbf{t}_\alpha\} \quad (7)$$

and for any \mathbf{k}

$$\sum_{i=1}^{\alpha} L_{\mathbf{k}}(\mathbf{t}_i) B_{\mathbf{t}_i} \geq - \begin{bmatrix} n \\ \mathbf{k} \end{bmatrix}. \quad (8)$$

Now, we may formulate the primal linear programming problem for the Lee-scheme [3]:

Theorem 1. *Let $B_{\mathbf{t}}^*$, $\mathbf{t} \in \{\mathbf{t}_1, \dots, \mathbf{t}_\alpha\}$ be an optimal solution of the linear programming problem*

$$\begin{aligned} \sum_{i=1}^{\alpha} B_{\mathbf{t}_i} &= \max! \\ B_{\mathbf{t}_i} &\geq 0, \quad i \in I \text{ and } B_{\mathbf{t}_i} = 0, \quad i \in \{1, \dots, \alpha\} \setminus I \\ \sum_{i=1}^{\alpha} B_{\mathbf{t}_i} L_{\mathbf{k}}(\mathbf{t}_i) &\geq - \begin{bmatrix} n \\ \mathbf{k} \end{bmatrix}, \end{aligned} \quad (9)$$

for all \mathbf{k} running through Lee-compositions,

where $I = \{i \mid l(\mathbf{x}) = \mathbf{t}_i \text{ and } w_L(\mathbf{x}) \geq d\}$. Then $1 + \sum_{i=1}^{\alpha} B_{\mathbf{t}_i}^*$ is an upper bound to the size of the code C with the minimum distance d .

3 Refinements on the Linear Programming Bound for Linear Lee-Codes

In the Hamming metric, multiplying codewords by some constant does not change the weight of the codewords. In the Lee metric, however, when a codeword is multiplied by some constant, the Lee-composition and the Lee-weight of the codeword are usually changed. Since linear codes are linear subspaces of vector spaces, all the vectors obtained by multiplying a codeword also belong to the code. In this section, we show that there must be as many codewords having the Lee-composition of a given codeword \mathbf{x} as there are codewords having the Lee-composition of the codeword $r\mathbf{x}$ that is obtained by multiplying \mathbf{x} by some constant r . This property can then be used to formulate constraining equalities into the linear programming problem, since the cardinality of a set of codewords having a given Lee-composition corresponds to a coefficient of the inner distribution of the code.

For simplicity, we let q be prime, $\mathbb{F}_q^n = \{\mathbf{x} \mid x_i \in \mathbb{F}_q, i = 1, \dots, n\}$. Let $C \subseteq \mathbb{F}_q^n$ be a linear code. For a linear code, if $\mathbf{x} \in C$, then $r\mathbf{x} \in C$ for all $r \in \{0, \dots, q-1\}$. Denote by $C_{\mathbf{t}}$ the set of codewords having the Lee-composition \mathbf{t} .

Let us first examine how to obtain the Lee-composition of the vector $r\mathbf{x}$ from the Lee-composition of the vector \mathbf{x} . The vector \mathbf{x} has the Lee-composition $l(\mathbf{x}) = [l_0(\mathbf{x}), l_1(\mathbf{x}), \dots, l_s(\mathbf{x})]$. The Lee-composition of the vector $r\mathbf{x}$ is clearly a permutation of the Lee-composition $l(\mathbf{x})$, since $r\mathbf{x}$ contains the same number of elements equal to rx_i as the vector \mathbf{x} contains elements equal to x_i . Thus, the Lee-composition of the vector $r\mathbf{x}$ is $l(r\mathbf{x}) = [l_{\pi_r(0)}(\mathbf{x}), l_{\pi_r(1)}(\mathbf{x}), \dots, l_{\pi_r(s)}(\mathbf{x})]$, where $\pi_r(i) = |k|$ such that $kr \equiv i \pmod q$ and $-s \leq k \leq s$.

Now, we introduce the following Lemma:

Lemma 1. *Given any Lee-composition \mathbf{t} and any integer $r \in \{1, \dots, q-1\}$, define the Lee-composition $\mathbf{u} = [t_0, t_{\pi_r(1)}, \dots, t_{\pi_r(s)}]$. For any linear code the sets $C_{\mathbf{t}}$ and $C_{\mathbf{u}}$ have equal cardinalities.*

Proof. First, any $\mathbf{x}_1 \in C_{\mathbf{t}}$ has one corresponding element $r\mathbf{x}_1 \in C_{\mathbf{u}}$. Also, every two distinct codewords in $C_{\mathbf{t}}$ correspond to two distinct elements in $C_{\mathbf{u}}$, since $r\mathbf{x}_1 \neq r\mathbf{x}_2$ if $\mathbf{x}_1 \neq \mathbf{x}_2$. Finally, every codeword in $C_{\mathbf{u}}$ corresponds to one codeword in $C_{\mathbf{t}}$, since for any $\mathbf{y} \in C_{\mathbf{u}}$, we may take r^{-1} such that $r^{-1}r \equiv 1 \pmod q$ and $r^{-1}\mathbf{y}$ will belong to $C_{\mathbf{t}}$. \square

The cardinalities of the sets $C_{\mathbf{t}}$ correspond to the coefficients of the inner distribution of the code C . Since there are equalities between these cardinalities, we get additional equality constraints in the linear programming problem for linear Lee-codes.

Let us denote by $\tau(\mathbf{t})$ the mapping that maps the Lee-composition \mathbf{t} into the set of Lee-compositions, which are obtained from \mathbf{t} by multiplication of vectors having the Lee-composition \mathbf{t} by all $r \in \{1, \dots, q-1\}$. Then,

$$\tau(\mathbf{t}) = \{[t_{\pi_r(0)}(\mathbf{x}), t_{\pi_r(1)}(\mathbf{x}), \dots, t_{\pi_r(s)}(\mathbf{x})] \mid \mathbf{t} = l(\mathbf{x}), \pi_r(i) = |k|, \\ kr \equiv i \pmod q, -s \leq k \leq s, r, k \in \mathbb{F}_q\}.$$

We may now formulate the linear programming problem for linear Lee-codes:

Theorem 2. Let $B_{\mathbf{t}}^*$, $\mathbf{t} \in \{\mathbf{t}_1, \dots, \mathbf{t}_\alpha\}$ be an optimal solution of the linear programming problem

$$\begin{aligned}
\sum_{i=1}^{\alpha} B_{\mathbf{t}_i} &= \max! \\
B_{\mathbf{t}_i} &\geq 0, \quad i \in I \text{ and } B_{\mathbf{t}_i} = 0, \quad i \in \{1, \dots, \alpha\} \setminus I \\
B_{\mathbf{t}_i} &= B_{\mathbf{t}_j} \text{ for all } \mathbf{t}_j \in \tau(\mathbf{t}_i) \\
\sum_{i=1}^{\alpha} B_{\mathbf{t}_i} L_{\mathbf{k}}(\mathbf{t}_i) &\geq - \begin{bmatrix} n \\ \mathbf{k} \end{bmatrix}, \\
&\text{for all } \mathbf{k} \text{ running through Lee-compositions,}
\end{aligned} \tag{10}$$

where $I = \{i \mid l(\mathbf{x}) = \mathbf{t}_i \text{ and } w_L(\mathbf{x}) \geq d\}$. Then $1 + \sum_{i=1}^{\alpha} B_{\mathbf{t}_i}^*$ is an upper bound to the size of the code C with the minimum distance d .

Let us look at an example on the equalities between different weight coefficients:

Example 1. Take the linear $[3, 2]$ -code over \mathbb{F}_7 with the generator matrix

$$G = \begin{bmatrix} 1 & 0 & 2 \\ 0 & 1 & 4 \end{bmatrix}.$$

The codewords of the code are

$$\begin{aligned}
&[0, 0, 0], [0, 1, 4], [0, 2, 1], [0, 3, 5], [0, 4, 2], [0, 5, 6], [0, 6, 3], \\
&[1, 0, 2], [1, 1, 6], [1, 2, 3], [1, 3, 0], [1, 4, 4], [1, 5, 1], [1, 6, 5], \\
&[2, 0, 4], [2, 1, 1], [2, 2, 5], [2, 3, 2], [2, 4, 6], [2, 5, 3], [2, 6, 0], \\
&[3, 0, 6], [3, 1, 3], [3, 2, 0], [3, 3, 4], [3, 4, 1], [3, 5, 5], [3, 6, 2], \\
&[4, 0, 1], [4, 1, 5], [4, 2, 2], [4, 3, 6], [4, 4, 3], [4, 5, 0], [4, 6, 4], \\
&[5, 0, 3], [5, 1, 0], [5, 2, 4], [5, 3, 1], [5, 4, 5], [5, 5, 2], [5, 6, 6], \\
&[6, 0, 5], [6, 1, 2], [6, 2, 6], [6, 3, 3], [6, 4, 0], [6, 5, 4], [6, 6, 1].
\end{aligned}$$

The different Lee-compositions of the codewords are

$$\begin{aligned}
&[3, 0, 0, 0], [1, 1, 1, 0], [1, 1, 0, 1], [1, 0, 1, 1], [0, 3, 0, 0], [0, 2, 1, 0], \\
&[0, 1, 1, 1], [0, 1, 0, 2], [0, 0, 3, 0], [0, 0, 2, 1], [0, 0, 0, 3].
\end{aligned}$$

Let us take, for example, the second Lee-composition $[1, 1, 1, 0]$. There are 6 codewords having this Lee-composition, $[0, 2, 1]$, $[0, 5, 6]$, $[1, 0, 2]$, $[2, 6, 0]$, $[5, 1, 0]$ and $[6, 0, 5]$. Therefore, the coefficient of the inner distribution corresponding to the Lee-composition $[1, 1, 1, 0]$ is 6. If we multiply these vectors by 2, we get the codewords $[0, 4, 2]$, $[0, 3, 5]$, $[2, 0, 4]$, $[4, 5, 0]$, $[3, 2, 0]$ and $[5, 0, 3]$, i.e., all such codewords, which have the Lee-composition $[1, 0, 1, 1]$. Similarly, by multiplying with other possible values we obtain sets of codewords corresponding to certain Lee-compositions.

In Table 1, the Lee-compositions \mathbf{t}_i of the code are listed together with the coefficients of the inner distribution $B_{\mathbf{t}_i}$. For each coefficient $B_{\mathbf{t}_i}$, the table shows also the set $\tau(\mathbf{t}_i)$ of those coefficients of the inner distribution, which are equal to $B_{\mathbf{t}_i}$ under the transformations following from the linearity of the code, i.e., the coefficients of the inner distribution corresponding to such Lee-compositions, which are obtained from the Lee-compositions \mathbf{t}_i by the mapping $\tau(\mathbf{t}_i)$. The Lee-compositions are indexed based on the lexicographic order of all Lee-compositions for \mathbb{F}_7^3 .

Table 1: The Lee-compositions \mathbf{t}_i , the coefficients $B_{\mathbf{t}_i}$ and the set $\tau(\mathbf{t}_i)$ of the $[3, 2]$ -code over \mathbb{F}_7 . The Lee-compositions are indexed based on the lexicographic order of all Lee-compositions for \mathbb{F}_7^3 .

i	\mathbf{t}_i	$B_{\mathbf{t}_i}$	$\tau(\mathbf{t}_i)$
0	(3,0,0,0)	1	$\{B_{\mathbf{t}_0}\}$
5	(1,1,1,0)	6	$\{B_{\mathbf{t}_5}, B_{\mathbf{t}_6}, B_{\mathbf{t}_8}\}$
6	(1,1,0,1)	6	$\{B_{\mathbf{t}_5}, B_{\mathbf{t}_6}, B_{\mathbf{t}_8}\}$
8	(1,0,1,1)	6	$\{B_{\mathbf{t}_5}, B_{\mathbf{t}_6}, B_{\mathbf{t}_8}\}$
10	(0,3,0,0)	2	$\{B_{\mathbf{t}_{10}}, B_{\mathbf{t}_{16}}, B_{\mathbf{t}_{19}}\}$
11	(0,2,1,0)	6	$\{B_{\mathbf{t}_{11}}, B_{\mathbf{t}_{15}}, B_{\mathbf{t}_{17}}\}$
14	(0,1,1,1)	6	$\{B_{\mathbf{t}_{14}}\}$
15	(0,1,0,2)	6	$\{B_{\mathbf{t}_{11}}, B_{\mathbf{t}_{15}}, B_{\mathbf{t}_{17}}\}$
16	(0,0,3,0)	2	$\{B_{\mathbf{t}_{10}}, B_{\mathbf{t}_{16}}, B_{\mathbf{t}_{19}}\}$
17	(0,0,2,1)	6	$\{B_{\mathbf{t}_{11}}, B_{\mathbf{t}_{15}}, B_{\mathbf{t}_{17}}\}$
19	(0,0,0,3)	2	$\{B_{\mathbf{t}_{10}}, B_{\mathbf{t}_{16}}, B_{\mathbf{t}_{19}}\}$

3.1 Dual Codes

The MacWilliams identities state that the coefficients of the inner distribution of the dual of a code are given by a transformations of the coefficients of the inner distribution of the original code (for further reading, see, for instance [14]). If we consider the above refinements with respect to the dual code, we may formulate more equality constraints, which follow from the connections between certain Lee-compositions in the dual code. Now [14],

$$\beta_{\mathbf{k}} = \frac{1}{|C|} \sum_{i=0}^{\alpha} L_{\mathbf{k}}(\mathbf{t}_i) B_{\mathbf{t}_i}, \quad (11)$$

where $\beta_{\mathbf{k}}$ is a coefficient of the inner distribution of the dual code. Since the dual code is linear, some of these coefficients must be equal to each other, i.e., for some $u \neq v$, $\beta_{\mathbf{k}_u} = \beta_{\mathbf{k}_v}$. Hence, we may write

$$\beta_{\mathbf{k}_u} = \frac{1}{|C|} \sum_{i=0}^{\alpha} L_{\mathbf{k}_u}(\mathbf{t}_i) B_{\mathbf{t}_i} = \frac{1}{|C|} \sum_{i=0}^{\alpha} L_{\mathbf{k}_v}(\mathbf{t}_i) B_{\mathbf{t}_i} = \beta_{\mathbf{k}_v}.$$

We get the equality constraints

$$\sum_{i=0}^{\alpha} L_{\mathbf{k}_u}(\mathbf{t}_i) B_{\mathbf{t}_i} - \sum_{i=0}^{\alpha} L_{\mathbf{k}_v}(\mathbf{t}_i) B_{\mathbf{t}_i} = \sum_{i=0}^{\alpha} (L_{\mathbf{k}_u}(\mathbf{t}_i) - L_{\mathbf{k}_v}(\mathbf{t}_i)) B_{\mathbf{t}_i} = 0.$$

Now, we may formulate the linear programming problem for linear Lee-codes with the equality constraints given by the dual code:

Theorem 3. *Let $B_{\mathbf{t}}^*$, $\mathbf{t} \in \{\mathbf{t}_1, \dots, \mathbf{t}_{\alpha}\}$ be an optimal solution of the linear programming problem*

$$\begin{aligned} \sum_{i=1}^{\alpha} B_{\mathbf{t}_i} &= \max! \\ B_{\mathbf{t}_i} &\geq 0, \quad i \in I \text{ and } B_{\mathbf{t}_i} = 0, \quad i \in \{1, \dots, \alpha\} \setminus I \\ \sum_{i=1}^{\alpha} B_{\mathbf{t}_i} L_{\mathbf{k}}(\mathbf{t}_i) &\geq - \begin{bmatrix} n \\ \mathbf{k} \end{bmatrix}, \\ &\text{for all } \mathbf{k} \text{ running through Lee-compositions,} \\ \sum_{i=0}^{\alpha} B_{\mathbf{t}_i} (L_{\mathbf{k}_u}(\mathbf{t}_i) - L_{\mathbf{k}_v}(\mathbf{t}_i)) &= 0 \text{ for all } \mathbf{k}_v \in \tau(\mathbf{k}_u), \end{aligned} \tag{12}$$

where $I = \{i \mid l(\mathbf{x}) = \mathbf{t}_i \text{ and } w_L(\mathbf{x}) \geq d\}$. Then $1 + \sum_{i=1}^{\alpha} B_{\mathbf{t}_i}^*$ is an upper bound to the size of the code C with the minimum distance d .

Theorem 4. *The problems in theorems 2 and 3 are equivalent.*

Proof. Rearrange now the indices of Lee-compositions so that in the sequence $\mathbf{t}_0, \dots, \mathbf{t}_{\alpha}$ the compositions $\tau(\mathbf{t})$ for which the coefficients $B_{\mathbf{t}}$ are constrained to have the same cardinalities will be in a consecutive order. Thus the valid solutions by the equality constraints given in Theorem 2 will be of the form

$$\begin{aligned} \begin{bmatrix} B_{\mathbf{t}_0} \\ B_{\mathbf{t}_1} \\ \vdots \\ B_{\mathbf{t}_{\alpha}} \end{bmatrix} &= \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ & & & & \ddots & \\ 0 & 0 & 0 & 0 & \cdots & 1 \end{bmatrix} \begin{bmatrix} \gamma_0 \\ \gamma_1 \\ \vdots \\ \gamma_{\kappa} \end{bmatrix} \\ \mathbf{B} &= A\gamma, \end{aligned}$$

where the matrix A has $\kappa + 1$ blocks, each having as many rows as there are elements in the corresponding set $\tau(\mathbf{t}_i)$.

By (11), for any linear code

$$\beta = \frac{1}{|C|} \Upsilon \mathbf{B},$$

where \mathbf{B} and β are the inner distributions of the code and its dual code and Υ is the matrix containing the Lee-numbers with $\Upsilon(\mathbf{t}, \mathbf{u}) = L_{\mathbf{t}}(\mathbf{u})$.

Since $\mathbf{B} = A\gamma$ and $\beta = A\gamma'$, we may write

$$A\gamma' = \frac{1}{|C|} \Upsilon A\gamma. \quad (13)$$

Now, we want to show that the above equation holds for any arbitrary γ in order to show the two problems equivalent. In other words, we want to show that the transformation of any vector of the form $A\gamma$ by the Lee-numbers Υ is a vector of the form $A\gamma'$.

Construct a linear code in \mathbb{F}_q^n by taking a generator matrix having just one vector with a Lee-composition \mathbf{t}_i . The code has an inner distribution with nonzero values at position $B_{\mathbf{t}_0}$ and positions $B_{\mathbf{t}_u}$, $\mathbf{t}_u \in \tau(\mathbf{t}_i)$. Hence, we obtain a vector γ_i with two nonzero values (with one nonzero value for the code having just the all-zero vector). Continue by taking another vector having a Lee-composition $\mathbf{t}_j \notin \tau(\mathbf{t}_i)$ as a generator matrix. Continue in such a way, for each Lee-composition not included in the previous sets τ , so that we obtain $\kappa + 1$ linearly independent vectors $\gamma_0, \dots, \gamma_\kappa$.

Any arbitrary vector γ can now be obtained as a linear combination of the linearly independent vectors $\gamma_0, \dots, \gamma_\kappa$, i.e., we may take $\gamma = a_0\gamma_0 + \dots + a_\kappa\gamma_\kappa$. Since, the vectors $\gamma_0, \dots, \gamma_\kappa$ are those of linear codes, the equality in (13) holds for them, i.e., we have for each γ_i ,

$$A\gamma'_i = \frac{1}{|C|} \Upsilon A\gamma_i. \quad (14)$$

We may then write

$$\begin{aligned} a_0 \cdot A\gamma'_0 + \dots + a_\kappa \cdot A\gamma'_\kappa &= a_0 \cdot \frac{1}{|C|} \Upsilon A\gamma_0 + \dots + a_\kappa \cdot \frac{1}{|C|} \Upsilon A\gamma_\kappa \\ A \cdot (a_0\gamma'_0 + \dots + a_\kappa\gamma'_\kappa) &= \frac{1}{|C|} \Upsilon A \cdot (a_0\gamma_0 + \dots + a_\kappa\gamma_\kappa) \\ A\gamma' &= \frac{1}{|C|} \Upsilon A\gamma. \end{aligned}$$

It remains to show that given any arbitrary $A\gamma'$, the equation (13) holds. We may construct γ' similarly as a linear combination of linearly independent vectors $\gamma'_0, \dots, \gamma'_\kappa$ constructed as the vectors $\gamma_0, \dots, \gamma_\kappa$ above. Hence, we may take $\gamma' = b_0\gamma'_0 + \dots + b_\kappa\gamma'_\kappa$. Again, for each γ'_i we have (14), and we may thus conclude using the above reasoning that the the equation (13) holds. □

4 Computational Aspects of the Linear Programming Problem for Lee-codes

In this section, we discuss the computational aspects of the linear programming problem. Accurate results in the linear programming problem depend on efficient and accurate computation. We introduce a recursive way for computing the Lee-numbers, which play an important part in the computations. Based on the refinements on the problem for linear codes, we introduce a more compact linear program, where the set of linear constraints is reduced based on the theory. We can perform all computations with integers, resulting in very accurate results.

4.1 Computing the Lee-numbers

In [19], a recursion for computing the Lee-numbers was introduced, providing the possibility of efficient computation of the bounds. We introduce here an alternative recursion following from the polynomial definition of the Lee-numbers. The recursion is based on the observation that as the length of the vector grows by one, it results in addition of 1 in some component of the Lee-composition depending on the added component.

The equations (2) and (3) give the Lee-numbers as coefficients of a generating polynomial. Let us now examine, how we can obtain them recursively using this generating polynomial by an example for $q = 5$. Denote $\xi = \exp(\frac{2\pi\sqrt{-1}}{5})$. The Lee-numbers $L_{\mathbf{t}}(\mathbf{u})$ are now given by

$$\begin{aligned} & (z_0 + 2z_1 + 2z_2)^{u_0} (z_0 + (\xi + \xi^{-1})z_1 + (\xi^2 + \xi^{-2})z_2)^{u_1} \cdot \\ & (z_0 + (\xi^2 + \xi^{-2})z_1 + (\xi + \xi^{-1})z_2)^{u_2} = \sum_{\mathbf{t}} L_{\mathbf{t}}(\mathbf{u}) z_0^{t_0} z_1^{t_1} z_2^{t_2}. \end{aligned} \quad (15)$$

Notice that if we denote $\zeta = \xi + \xi^{-1} = \xi + \xi^4$, then $\zeta^2 = \xi^2 + \xi^{-2} + 2\xi\xi^{-1} = \xi^2 + \xi^3 + 2$. Because $1 + \xi + \xi^2 + \xi^3 + \xi^4 = 0$ we have $\zeta^2 + \zeta - 1 = 0$. Therefore, $\xi^2 + \xi^{-2} = -1 - \zeta$, and we can write (15) as

$$\begin{aligned} & (z_0 + 2z_1 + 2z_2)^{u_0} (z_0 + \zeta z_1 + (-1 - \zeta)z_2)^{u_1} \cdot \\ & (z_0 + (-1 - \zeta)z_1 + \zeta z_2)^{u_2} = \sum_{\mathbf{t}} L_{\mathbf{t}}(\mathbf{u}) z_0^{t_0} z_1^{t_1} z_2^{t_2}. \end{aligned} \quad (16)$$

Assume that we have the Lee-numbers $L_{\mathbf{t}}(u_0, u_1, u_2)$. Then for $L_{\mathbf{t}}(u_0 + 1, u_1, u_2)$ we have

$$\begin{aligned} & \sum_{\mathbf{t}} L_{\mathbf{t}}(u_0 + 1, u_1, u_2) z_0^{t_0} z_1^{t_1} z_2^{t_2} \\ &= (z_0 + 2z_1 + 2z_2) \sum_{\mathbf{t}} L_{\mathbf{t}}(u_0, u_1, u_2) z_0^{t_0} z_1^{t_1} z_2^{t_2} \\ &= L_{(t_0-1, t_1, t_2)}(u_0, u_1, u_2) z_0^{t_0} z_1^{t_1} z_2^{t_2} + 2L_{(t_0, t_1-1, t_2)}(u_0, u_1, u_2) z_0^{t_0} z_1^{t_1} z_2^{t_2} + \\ & 2L_{(t_0, t_1, t_2-1)}(u_0, u_1, u_2) z_0^{t_0} z_1^{t_1} z_2^{t_2}. \end{aligned}$$

Therefore,

$$\begin{aligned} L_{\mathbf{t}}(u_0 + 1, u_1, u_2) &= L_{(t_0-1, t_1, t_2)}(u_0, u_1, u_2) + \\ &2L_{(t_0, t_1-1, t_2)}(u_0, u_1, u_2) + 2L_{(t_0, t_1, t_2-1)}(u_0, u_1, u_2). \end{aligned}$$

Similarly, for $L_{\mathbf{t}}(u_0, u_1 + 1, u_2)$ and $L_{\mathbf{t}}(u_0, u_1, u_2 + 1)$ we have

$$\begin{aligned} L_{\mathbf{t}}(u_0, u_1 + 1, u_2) &= L_{(t_0-1, t_1, t_2)}(u_0, u_1, u_2) + \\ &\zeta L_{(t_0, t_1-1, t_2)}(u_0, u_1, u_2) + (-1 - \zeta) L_{(t_0, t_1, t_2-1)}(u_0, u_1, u_2), \end{aligned}$$

and

$$\begin{aligned} L_{\mathbf{t}}(u_0, u_1, u_2 + 1) &= L_{(t_0-1, t_1, t_2)}(u_0, u_1, u_2) + \\ &(-1 - \zeta) L_{(t_0, t_1-1, t_2)}(u_0, u_1, u_2) + \zeta L_{(t_0, t_1, t_2-1)}(u_0, u_1, u_2). \end{aligned}$$

Notice, that the above recursions are of the form

$$L_1 = L_2 + (a + b\zeta)L_3 + (c + d\zeta)L_4,$$

and the possible initial values for $q = 5$ with $n = 1$ are exactly the coefficients appearing in these recursions, $\{1, 2, \zeta, -1 - \zeta\}$. Because

$$(a + b\zeta)(c + d\zeta) = ac + (ad + bc)\zeta + bd\zeta^2 = ac + bd + (ad + bc - bd)\zeta,$$

we see that if we represent the Lee-numbers as vectors $[a, b]$ and define multiplication as $[a, b] \cdot [c, d] = [ac + bd, ad + bc + bd]$, we can perform all calculations with integers.

Consider now the case for $q = 7$, where $\xi = \exp(\frac{2\pi\sqrt{-1}}{7})$. Denote again by $\zeta = \xi + \xi^{-1}$. Again, $\zeta^2 = \xi^2 + \xi^{-2} + 2\xi\xi^{-1} = \xi^2 + \xi^3 + 2$. For $q = 7$, $1 + \xi + \xi^2 + \xi^3 + \xi^4 + \xi^5 + \xi^6 = 0$, so we get for $\zeta^3 = -\zeta^2 + 2\zeta + 1$. The Lee-numbers $L_{\mathbf{t}}(\mathbf{u})$ are now given by

$$\begin{aligned} &(z_0 + 2z_1 + 2z_2 + 2z_3)^{u_0} (z_0 + \zeta z_1 + (\zeta^2 - 2)z_2 + (-\zeta^2 - \zeta + 1)z_3)^{u_1} \cdot \\ &(z_0 + (\zeta^2 - 2)z_1 + (-\zeta^2 - \zeta + 1)z_2 + \zeta z_3)^{u_2} \cdot \\ &(z_0 + (-\zeta^2 - \zeta + 1)z_1 + \zeta z_2 + (\zeta^2 - 2)z_3)^{u_3} = \sum_{\mathbf{t}} L_{\mathbf{t}}(\mathbf{u}) z_0^{t_0} z_1^{t_1} z_2^{t_2}. \end{aligned}$$

Hence, the recursions will be of the form

$$L_1 = L_2 + (a + b\zeta + c\zeta^2)L_3 + (d + e\zeta + f\zeta^2)L_4 + (g + h\zeta + i\zeta^2)L_5.$$

The multiplication of two coefficients in the above equation is

$$\begin{aligned} &(a + b\zeta + c\zeta^2)(d + e\zeta + f\zeta^2) \\ &= ad + ae\zeta + af\zeta^2 + bd\zeta + be\zeta^2 + bf\zeta^3 + cd\zeta^2 + ce\zeta^3 + cf\zeta^4, \end{aligned}$$

which, since $\zeta^4 = 3\zeta^2 - \zeta - 1$, results in the multiplication rule $[a, b, c] \cdot [d, e, f] = [ad + bf + ce - cf, ae + bd + 2ce + 2bf - cf, af + be + cd - bf - ce + 3cf]$ and we may again perform all computations with integers.

For the general q , the powers of ζ will be reduced according to the cyclotomic polynomial $\Phi_q(\xi)$, where ξ are the roots of the cyclotomic polynomial, i.e., the primitive roots of unity $\xi = \exp(\frac{2\pi\sqrt{-1}}{q})$. This recursion provides very accurate values for the Lee-numbers, resulting in more accurate optimization in the linear programming problem.

4.2 Compacting the Set of Linear Constraints

Most linear programming solvers allow to express the constraints of the problems both in terms of inequality and equality constraints, thus the two formulations of the LP problem given in Theorems 2 and 3 can easily be programmed and run. We examine the structure of the problem so that we can formulate it in a more compact form, leading to a faster execution.

We notice that by replacing the α variables $B_{t_1}, \dots, B_{t_\alpha}$ of the LP problem with the set of variables $\gamma_1, \dots, \gamma_\kappa$ we are eliminating the equality constraints from the LP problem. We introduce the vector $\boldsymbol{\gamma} = (\gamma_0, \gamma_1, \dots, \gamma_\kappa)$ and formulate the equivalent LP problem:

$$\begin{aligned} & \max_{\gamma_1, \dots, \gamma_\kappa} \sum_{i=1}^{\kappa} |\tau_i| \gamma_i \\ & \text{subject to} \\ & \gamma_0 = 1 \text{ and } \gamma_i \geq 0, i \in I \text{ and } \gamma_i = 0, i \in \{1, \dots, \alpha\} \setminus I \\ & \Upsilon A \boldsymbol{\gamma} \geq 0 \end{aligned}$$

The cardinalities $|\tau_i|$ appear in the criterion of the problem since the initial criterion expressed in \mathbf{B} is $\mathbf{1}^T \mathbf{B}$ (where $\mathbf{1}$ is the all one vector), and the criterion in the new variables is $\mathbf{1}^T A \boldsymbol{\gamma}$, where the new vector of coefficients, $\mathbf{1}^T A$, will have as elements the size of the partitions of A , which are equal to the cardinalities of the sets τ_i .

Additionally we notice that the matrix $\mathcal{U} = \Upsilon A$ can be seen to have the partition structure similar to that of A ,

$$\begin{aligned} \mathcal{U} = \Upsilon A = \Upsilon & \begin{bmatrix} 1 & 0 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & 0 & \dots & 0 \\ 0 & 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & 0 & \dots & 0 \\ \dots & & & & & \\ 0 & 0 & 0 & 0 & \dots & 1 \end{bmatrix} = \begin{bmatrix} \Phi_{1,1} & \Phi_{1,2} & \Phi_{1,3} & \Phi_{1,4} & \dots & \Phi_{1,\kappa} \\ \Phi_{2,1} & \Phi_{2,2} & \Phi_{2,3} & \Phi_{2,4} & \dots & \Phi_{2,\kappa} \\ \Phi_{2,1} & \Phi_{2,2} & \Phi_{2,3} & \Phi_{2,4} & \dots & \Phi_{2,\kappa} \\ \Phi_{2,1} & \Phi_{2,2} & \Phi_{2,3} & \Phi_{2,4} & \dots & \Phi_{2,\kappa} \\ \Phi_{2,1} & \Phi_{2,2} & \Phi_{2,3} & \Phi_{2,4} & \dots & \Phi_{2,\kappa} \\ \Phi_{3,1} & \Phi_{3,2} & \Phi_{3,3} & \Phi_{3,4} & \dots & \Phi_{3,\kappa} \\ \Phi_{3,1} & \Phi_{3,2} & \Phi_{3,3} & \Phi_{3,4} & \dots & \Phi_{3,\kappa} \\ \dots & & & & & \\ \Phi_{\kappa,1} & \Phi_{\kappa,2} & \Phi_{\kappa,3} & \Phi_{\kappa,4} & \dots & \Phi_{\kappa,\kappa} \end{bmatrix} \\ & = A \Phi \end{aligned}$$

where the matrix $\Phi = [\Phi_{i,j}]$ is $\kappa \times \kappa$ and the rows in a partition corresponds to the Lee compositions belonging to the same set τ_i . Inside a partition the rows of the matrix \mathcal{U} are identical, leading to a repeated inequality constraint. In order to remove this redundancy, we are selecting from the matrix \mathcal{U} only one row per partition block, keeping thus only the non-redundant inequalities, resulting in a matrix \mathcal{U}' , and we replace the inequality constraints $\Upsilon A \boldsymbol{\gamma} \geq 0$ with $\mathcal{U}' \boldsymbol{\gamma} \geq 0$.

We also observe that the matrix \mathcal{U} is formed of integer numbers, as opposed to the matrix Υ where the elements are various combinations of the powers of ξ , and are in most cases irrational numbers. This can be seen as follows.

The elements of \mathcal{U} are sums of k Lee-numbers:

$$\begin{aligned} & L_{\mathbf{t}_1}(\mathbf{u}) + L_{\mathbf{t}_2}(\mathbf{u}) + \cdots + L_{\mathbf{t}_k}(\mathbf{u}) \\ = & \sum_{\mathbf{x}|L(\mathbf{x})=\mathbf{t}_1} \left(\prod_{i=1}^n \xi^{v_i x_i} \right) + \sum_{\mathbf{x}|L(\mathbf{x})=\mathbf{t}_2} \left(\prod_{i=1}^n \xi^{v_i x_i} \right) + \cdots + \sum_{\mathbf{x}|L(\mathbf{x})=\mathbf{t}_k} \left(\prod_{i=1}^n \xi^{v_i x_i} \right) \end{aligned} \quad (17)$$

where k is the cardinality of $\tau(\mathbf{t}_i)$.

Since the Lee-numbers $L_{\mathbf{t}_i}(\mathbf{u})$ correspond to compositions according to τ , then for each vector \mathbf{x} , in (17) are also included all the vectors $r\mathbf{x}$, where $r \in \{1, \dots, q-1\}$. Also, each vector can only have one Lee-composition and thus appear in only one of the sums in (17).

We may now rearrange and group the sum in (17) according to these multiplications:

$$(\xi^{\mathbf{v} \cdot \mathbf{x}_1} + \xi^{\mathbf{v} \cdot 2\mathbf{x}_1} + \cdots + \xi^{\mathbf{v} \cdot (q-1)\mathbf{x}_1}) + (\xi^{\mathbf{v} \cdot \mathbf{x}_2} + \xi^{\mathbf{v} \cdot 2\mathbf{x}_2} + \cdots + \xi^{\mathbf{v} \cdot (q-1)\mathbf{x}_2}) + \cdots$$

This forms a partition of the set of vectors having a Lee-composition in $\tau(\mathbf{t}_i)$, since the relation \mathcal{R} defined as $(\mathbf{x}, \mathbf{y}) \in \mathcal{R}_x$ iff $\mathbf{x} = r\mathbf{y}$, $r \in \{1, \dots, q-1\}$ is clearly an equivalence relation.

Therefore, we can group the sum into m parts, each having $q-1$ terms. If we now look at one such part:

$$\xi^{\mathbf{v} \cdot \mathbf{x}_i} + \xi^{\mathbf{v} \cdot 2\mathbf{x}_i} + \cdots + \xi^{\mathbf{v} \cdot (q-1)\mathbf{x}_i},$$

we see that, since $\mathbf{x} \cdot r\mathbf{y} = r(\mathbf{x} \cdot \mathbf{y})$, we have

$$\xi^{\mathbf{v} \cdot \mathbf{x}_i} + \xi^{2(\mathbf{v} \cdot \mathbf{x}_i)} + \cdots + \xi^{(q-1)\mathbf{v} \cdot \mathbf{x}_i},$$

where each exponent in the above sum is distinct. If the dot product $\mathbf{v} \cdot \mathbf{x}_i$ is 0, the above sum is $q-1$. Otherwise, we have a sum of the form

$$\xi + \xi^2 + \cdots + \xi^{q-1} = -1.$$

So (17) is a sum of the form $m_1(q-1) + m_2(-1)$, where m_1 and m_2 are integers ≥ 0 such that $m = m_1 + m_2$.

We may now use the more compact version of the linear programming problem performing computations only on integers, obtaining faster execution and very accurate results.

5 Results

In Tables 2-3 are the results for the upper bound of the parameter k for linear Lee-codes with $q = 5$ and $q = 7$, obtained using the linear programming bound for linear codes. The most interesting cases are the situations where the general linear programming bound would allow for a linear code to exist with some parameter k but the refinement shows that such a code cannot exist. For example, with $q = 5$, $n = 8$ and $d = 8$ the linear programming bound is 134 [19], which does not deny the existence of a linear code with $k = 3$. However, the refinement gives a bound of 75, which shows that there cannot be a code with the parameter $k = 3$.

Another example would be for $q = 7$ when $n = 7$ and $d = 11$. The linear programming bound gives a bound of 55, but with the refinement for linear codes the value 40 is obtained, implying that a linear code with $k = 2$ cannot exist with these parameters.

In Tables 2-3 bounds that were found to be tight are also shown. This was concluded by checking the minimum distances of linear codes generated randomly with given parameters q , n and k .

Table 2: Upper bounds for the parameter k of linear Lee-codes when $q = 5$. The * indicates a tight bound and bold an improvement comparing to linear programming bounds in [19].

$n \backslash d$	3	4	5	6	7	8	9	10	11	12	13	14-15
2	1*											
3	1*	1*										
4	2*	2*	1*	1*								
5	3*	3*	2*	1*	1*							
6	4*	3*	3*	2*	1*	1*	1*					
7	5*	4*	3*	3*	2*	1*	1*	1*				
8	6*	5*	4*	4*	3*	2*	2*	1*	1*	1*		
9	7*	6*	5*	5	4	3*	3	2*	1*	1*	1*	
10	8*	7*	6*	6	5	4	3*	3	2*	2*	1*	1*

Table 3: Upper bounds for the parameter k of linear Lee-codes when $q = 7$. The * indicates a tight bound and bold an improvement comparing to linear programming bounds in [19].

$n \backslash d$	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
2	1*															
3	2*	1*	1*	1*												
4	2*	2*	2*	1*	1*											
5	3*	3*	2*	2*	1*	1*	1*									
6	4*	4*	3*	3*	2*	2*	1*	1*	1*	1*						
7	5*	5*	4*	4	3*	3	2*	2*	1*	1*	1*					
8	6*	6*	5*	4*	4	4	3*	3	2*	2*	1*	1*	1*			
9	7*	6*	6	5*	5	4*	4	3*	3	3	2*	1*	1*	1*	1*	1*

5.1 Codes meeting the Bounds

In the following, we give some examples of codes, that meet the bounds obtained for linear codes.

Consider the bound for k given in Table 2 with $q = 5$, $n = 8$ and $d = 8$, which is 2. This means that the maximum number of codewords in a linear code with these parameters is at

most 25. The code having the generator matrix

$$G_1 = \begin{bmatrix} 1 & 0 & 0 & 2 & 2 & 3 & 3 & 1 \\ 0 & 1 & 2 & 3 & 0 & 3 & 4 & 3 \end{bmatrix}$$

is a $[8, 2]$ -code with the minimum distance 8, therefore, it is an optimal linear code for the above parameters.

Consider the bound for k given in Table 2 with $q = 5$, $n = 9$ and $d = 5$, which is 5. This means that the maximum number of codewords in a linear code with these parameters is at most 3125. The code having the generator matrix

$$G_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 4 & 2 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 2 & 4 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 2 & 2 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 3 & 3 & 3 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 2 & 2 & 2 \end{bmatrix}$$

is a $[9, 5]$ -code with the minimum distance 5, therefore, it is an optimal linear code for the above parameters.

Consider the bound for k given in Table 3 with $q = 7$, $n = 7$ and $d = 5$, which is 4. This means that the maximum number of codewords in a linear code with these parameters is at most 2401. The code having the generator matrix

$$G_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 5 & 4 & 4 \\ 0 & 1 & 0 & 0 & 3 & 6 & 6 \\ 0 & 0 & 1 & 0 & 1 & 4 & 6 \\ 0 & 0 & 0 & 1 & 6 & 5 & 3 \end{bmatrix}$$

is a $[7, 4]$ -code with the minimum distance 5, therefore, it is an optimal linear code for the above parameters.

6 Conclusions

In this paper, we introduced refinements on the linear programming bound for linear Lee-codes. These refinements are based on the observation that in the Lee metric, the multiplication of codewords typically changes the Lee-composition of the codeword and so also usually the Lee-weight. Therefore, and since the codes are linear, we are able to generate a mapping between the Lee-compositions, which follows in equalities between the coefficients of the inner distribution of the code, and, thus, in additional equality constraints in the linear programming problem. This refinement results in tighter bounds for linear Lee-codes.

We also discussed the computational aspects of the linear programming problem, including the computation of Lee-numbers, which can be done by recursion, and introduced an effective recursion based on the polynomial representation of the Lee-numbers. We introduced also a more compact form of the linear programming problem based on the refinements on linear Lee-codes. Our method is very accurate, since it allows all computations to be performed with integers.

References

- [1] C. Lee, “Some properties of nonbinary error-correcting codes,” *IRE Transactions on Information Theory*, vol. 4, no. 2, pp. 77–82, June 1958.
- [2] S. W. Golomb and L. R. Welch, “Perfect codes in the Lee metric and the packing of polyominoes,” *SIAM Journal on Applied Mathematics*, vol. 18, no. 2, pp. 302–317, 1970.
- [3] J. Astola, “The theory of Lee-codes,” Lappeenranta University of Technology, Department of Physics and Mathematics, Research Report 1/1982.
- [4] P. Horak, “On perfect Lee codes,” *Discrete Mathematics*, vol. 309, no. 18, pp. 5551–5561, September 2009.
- [5] C. Araujo, I. Dejter, and P. Horak, “A generalization of Lee codes,” *Designs, Codes and Cryptography*, vol. 70, no. 1-2, pp. 77–90, 2014.
- [6] T. Etzion, A. Vardy, and E. Yaakobi, “Dense error-correcting codes in the Lee metric,” in *Information Theory Workshop (ITW), 2010 IEEE*, September 2010, pp. 1–5.
- [7] T. Etzion, “Product constructions for perfect Lee codes,” *IEEE Transactions on Information Theory*, vol. 57, no. 11, pp. 7473–7481, November 2011.
- [8] E. R. Berlekamp, *Algebraic Coding Theory*. New York: McGraw-Hill, 1968.
- [9] R. Roth and P. Siegel, “Lee-metric BCH codes and their application to constrained and partial-response channels,” *Information Theory, IEEE Transactions on*, vol. 40, no. 4, pp. 1083–1096, July 1994.
- [10] Y. Wu and C. Hadjicostis, “Decoding algorithm and architecture for BCH codes under the Lee metric,” *IEEE Transactions on Communications*, vol. 56, no. 12, pp. 2050–2059, December 2008.
- [11] H. Astola and S. Stanković, “On the use of Lee-codes for constructing multiple-valued error-correcting decision diagrams,” in *5th International Symposium on Communications, Control, and Signal Processing*, Rome, Italy, May 2-4, 2012.
- [12] P. Delsarte, “An algebraic approach to the association schemes of coding theory,” *Philips Res. Repts. Suppl.*, 1973.
- [13] P. Delsarte and V. Levenshtein, “Association schemes and coding theory,” *IEEE Transactions on Information Theory*, vol. 44, no. 6, pp. 2477–2504, October 1998.
- [14] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*. Amsterdam: North-Holland, 1997.

- [15] R. McEliece, E. Rodemich, H. Rumsey, and L. Welch, “New upper bounds on the rate of a code via the Delsarte-MacWilliams inequalities,” *IEEE Transactions on Information Theory*, vol. 23, no. 2, pp. 157–166, March 1977.
- [16] E. Byrne, M. Greferath, and M. OSullivan, “The linear programming bound for codes over finite Frobenius rings,” *Designs, Codes and Cryptography*, vol. 42, no. 3, pp. 289–301, 2007.
- [17] M. Grassl, “Bounds on the minimum distance of linear codes and quantum codes,” Online available at <http://www.codetables.de>, 2007.
- [18] J. Quistorff, “New upper bounds on Lee codes,” *Discrete Applied Mathematics*, vol. 154, no. 10, pp. 1510 – 1521, 2006.
- [19] H. Astola and I. Tabus, “Bounds on the size of Lee-codes,” in *8th International Symposium on Image and Signal Processing and Analysis*, Trieste, Italy, September 2013, pp. 464–469, <http://www.cs.tut.fi/%7Eastola/0091-0144.pdf>.
- [20] H. Astola, “Bounds on the size of Lee-codes,” Online available at <http://www.cs.tut.fi/%7Eastola/leecodetables.html>, 2013.